Adaptive Sampling of Biomedical Images with Cartesian Genetic Programming

Yuri Lavinas[1(✉)], Nathan Haut[2], William Punch[2], Wolfgang Banzhaf[2], and Sylvain Cussat-Blanc[1]

[1] IRIT - CNRS UMR5505, University of Toulouse, Toulouse, France
lavinas.yuri.xp@alumni.tsukuba.ac.jp, sylvain.cussat-blanc@irit.fr
[2] Michigan State University, East Lansing, USA
{hautnath,punch,banzhafw}@msu.edu

Abstract. In this contribution we study how to effectively evolve programs tailored for biomedical image segmentation by using an Active Learning approach in Cartesian Genetic Programming (CGP). Active Learning allows to dynamically select training data by identifying the most informative next image to add to the training set. We study how different metrics for selecting images under active learning impact the searchability of CGP. Our results show that datasets built during evolution with active learning improve the performance of Cartesian GP substantially. In addition, we found that the choice of the particular metric used for selecting which images to add heavily impacts convergence speed. Our work shows that the right choice of the image selection metric positively impacts the effectiveness of the evolutionary algorithm.

Keywords: cartesian genetic programming · biomedical data · data sampling · active learning

1 Introduction

The field of biomedical image analysis has experienced a significant revolution with the usage of Neural Networks, particularly Deep Learning (DL) techniques, displaying a similar trend that happened in the broader domain of Computer Vision. Deep Neural Networks exhibit remarkable performance across various image classification and segmentation tasks within medical disciplines such as dermatology, radiology, and pathology [14,15], occasionally surpassing human experts. Nonetheless, DL methodologies encounter two primary limitations. Firstly, they are regarded as black-box systems, as the decision-making processes of deep Neural Networks often lack interpretability, a difficult challenge in critical domains like medicine. Secondly, DL approaches necessitate substantial amounts of annotated data, a resource-intensive and costly task typically carried out by busy experts, that reduces the acquisition of larger datasets for training purposes.

Recent work has demonstrated that Cartesian Genetic Programming (CGP) is an effective approach to address the above-mentioned limitations inherent in DL [12,28], with results competitive to DL. CGP evolves solutions based on a

M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15148, pp. 256–272, 2024.
https://doi.org/10.1007/978-3-031-70055-2_16

set of mathematical functions that process given inputs to produce an expected output. The phenotype of a CGP program can be represented as a graph which often is evolved using a $(1+\lambda)$ evolutionary strategy [31]. One of the main benefits of CGP is the use of a fixed-length integer-based genome to encode the functional graphs, mitigating the bloat effect encountered in many tree GP approaches, allows evolution to design small and interpretable solutions [40]. Particularly in image processing tasks, the CGP function library contains Computer Vision (CV) functions that are combined and optimized in order to construct tailored image processing pipelines suitable for specific objectives.

Minimizing the amount of necessary data for evolving effective programs in CV tasks poses an important challenge. Our goal in this work is to improve the evolution of CGP when processing biomedical images. In our previous work [29] we showed that building a small set of data samples in a dataset increases the convergence speed of CGP and leads to potentially more diverse solutions.

It is now well established Active Learning (AL) is able to effectively sample the most informative data points on different domains of knowledge [16,17,20,34]. In specific in Image Processing tasks [10,26,29], AL methods can provide performance improvements by using information from evolution (or training) to iteratively build a dataset. The main idea is to identify useful sample images from a larger dataset to apply during training which create programs of equivalent performance, but with less overhead [11]. Also, AL might help evolution avoid overfitting [6].

We here argue that we can incorporate ideas from AL methodologies and apply them to CGP to generate more data-efficient programs for biomedical image segmentation. We focus our efforts on developing a method to sample informative images, selected during evolution, resulting in a smaller training set than without these methods. We expect that CGP with AL can achieve higher performance with faster convergence than traditional CGP. This work extends our recent works [29,30] by an in-depth discussion of the impacts of Active Learning in CGP, including a discussion the frequency of images sampled and the potential impacts of the highly frequent images in the search dynamics of CGP; investigating the experimental results in terms of performance; convergence speed; program size; and on how to use information about the frequency of sampled images to improve CGP performance. Therefore, our contributions can be summarized as follows:

(1) We show that some images lead to more uncertainty in the number parallel runs of CGP variants and discuss possible reasons why.
(2) We study the sampling frequency of each image available to be used in the training dataset.
(3) We investigate if using information about the frequency of sampling can directly affect the performance of CGP.

The paper is organized as follows: Section 2 introduces the necessary background. Section 3 explains relevant concepts. Section 4 gives the experimental setup. Section 5 presents the experimental results of our analysis. Finally, Sect. 6 concludes the paper and discusses further research.

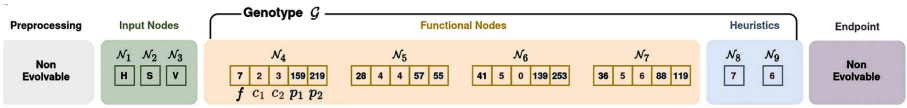


Fig. 1. The genotype of CGP.

Table 1. Description of the function library used in CGP.

Function	Arity	Function	Arity
Max	2	Min	2
Mean	2	Add	2
Subract	2	Bitwise_not	1
Bitwise_or	2	Bitwise_and	2
Bitwise_and_mask	2	Bitwise_xor	2
sqrt	1	pow2	1
exp	1	log	1
median_blur	1	gaussian_blur	1
laplacian	1	sobel	1
robert_cross	1	canny	1
sharpen	1	gabor	1
abs_diff	1	abs_diff2	2
fluo_tophat	1	ref_diff	1
erode	1	dilate	1
open	1	close	1
morph_gradient	1	morph_tophat	1
morph_blackhat	1	fill_holes	1
remove_small_objects	1	remove_small_holes	1
threshold	1	threshold_at_1	1
distance_transform	1	distance_transform_and_thresh	1
inrange_bin	1	inrange	1

2 Preliminaries

Most work in the area of Computer Vision tasks uses blackbox approaches, such as artificial Deep Neural Networks. However, these models have proven to be difficult for humans to analyze and interpret [12,33]. One way to complement these blackbox methods is to use methods that are inherently explainable. The class of Genetic Programming (GP) approaches is among such methods. Here we highlight Cartesian Genetic Programming (CGP), an evolutionary computation algorithm that evolves easier-to-interpret programs (as with most GP variants), in comparison to Deep Learning models [1,9,12,18,36,40].

2.1 Cartesian Genetic Programming

Cartesian Genetic Programming (CGP) is a Genetic Programming variant [31] specialized in evolving graph genotypes. Such graphs are often direct and acyclic and are indexed by Cartesian coordinates. Evolution defines how to connect the

nodes of the graphs and the function of each node. CGP has been successfully applied in multiple domains [1,9,36]. Specifically, CGP has been applied in Computer Vision tasks such as for controlling agents to play ATARI games [40], and in image processing tasks, such as biomedical image segmentation and object detection in robotics [12,18].

CGP generally employs the (1+λ) Evolutionary Algorithm (EA), although any other evolutionary algorithm could be used. Initially, a population of λ individuals is randomly generated and evaluated on the problem in question. Then, evaluation is conducted by first generating the programs from the graphs and then measuring their performance on the task considered. The solution with the highest performance is maintained to the next generation step, influencing the next λ individuals created via mutation. This process is repeated until a termination criterion is reached. For more information, see [9,31,40].

GP has been widely used in the biomedical domain, and as Khan et al. stated, GP is usually used in classification of cancerous cells [25]. In particular, one can find GP contributions that involve feature extraction [3,4,8,24,38] or that involve image classification [2,5,41].

2.2 Dynamic Data Sampling

Dynamic data sampling is one of the most important components of Active Learning (AL) methods and its effects have been studied on different Machine Learning algorithms [13,20,37]. AL is frequently used in Deep Learning in the context of processing biomedical data [16,32] with most of the work on AL focusing on finding the metric that leads models to the highest performance [16]. Interestingly, there are papers that suggest random sampling as a strong baseline [10,26].

In the domain of GP, the efficacy of Active Learning for symbolic regression tasks has been widely demonstrated in a diverse group of research, from works that focus on reducing the number of such evaluations [17] and on creating smaller, balanced datasets by recursively keeping the most 'meaningful' exemplars [39] to studies on improving the rate and consistency at which well-performing solutions are found while reducing the required number of training samples [19,20]. For classification problems, Hamida et al. [6,22] showed how different sampling methods studied across the years affect the performance of GP. Yet, we did not find works that combine AL and GP in the biomedical domain. Recently, we have shown that AL [29,30] methods can provide performance improvements in bio-medical image processing.

Here, we use uncertainty as the base for dynamic data sampling, in a similar manner as discussed by Nguyen et al. [23]. That is, given a model trained in a dataset D, each image not in the training dataset is assigned an uncertainty value based on the model's behavior. The image with highest uncertainty is labelled (by an oracle or expert) and added to D. Then, D is given to the model for training.

Algorithm 1. Adaptive Bio-image Sampling in CGP (ABS-CGP)

1: Initialize $(1 + \lambda)$ CGP.
2: **while** *Stopping criterion is not met* **do**
3: **Evolve** CGP with the training dataset.
4: **Update** elite solution.
5: **Calculate** uncertainty *metric* on the images not in the training set.
6: **Add** one image to the training dataset, the image with the highest uncertainty *metric* value.
7: **end while**

2.3 CGP Implementation

The CGP implementation we use is based on [12], a modular Cartesian Genetic Programming system to generate programs for computer vision tasks. This system introduces the notion of non-evolvable nodes which are functions not subjected to optimization in the syntactic graph. Through this algorithm, the image processing stack optimizes the order of functions and their parameters resulting in an image processing algorithm similar to a human-designed one, since they are based on established functions for image processing.

Figure 1 shows how our CGP implementation works. This genotype is a sequence of integers known as genes (each one represented as a box containing a single integer) that are organized into nodes. A node is composed of a single function drawn from the function library, at least one connection, and optional parameters. Moreover, in this implementation, the end nodes of CGP are connected to a fixed endpoint. This endpoint is useful since it allows CGP to obtain insight given by a Computer Vision expert. We use the Watershed Transform [7] as the endpoint. Thus, our CGP has 2 outputs, corresponding to the mask and markers necessary for the Watershed Transform endpoint. Finally, we use image processing functions mostly from OpenCV and Skimage, which apply programs directly to images. Only a few nodes, the "active" nodes, are used, as they are actually connected to the output of the program graph. Other nodes with no connections to the output are called "inactive" nodes. The outputs of a program can come from any node and are determined by the evolutionary process.

Table 1 lists the basic functions used in our function library and their related arities. These operators are functions from the OpenCV Python package and are fixed for all CGP variants compared[1].

3 Adaptive Image Sampling in Cartesian GP

The CGP with **A**daptive **B**iomedical-image **S**ampling (ABS-CGP) template we propose for instantiating and designing sampling method variants is shown in Algorithm 1. The main difference to standard CGP is that instead of using a

[1] The number of parameters needed by a given function.

fixed training dataset during evolution, our template uses a smaller dataset that is selected during evolution, given the different metrics (explained below). We sample the subset of images to be part of the training dataset used in evolution via sampling mechanisms. Two of these mechanisms sample data based on uncertainty-related information, while the other mechanism samples images based on values taken from a uniform distribution.

To calculate uncertainty in CGP our (1+λ) EA[2], we use a group of parallel CGP runs, executed in parallel. Uncertainty-based AL utilizes this group of diverse runs of programs to search different areas of the search space, the diversity of the models then allows their disagreement to be used as an uncertainty measure to select new training data where uncertainty is maximized. The idea is that selecting data where uncertainty is high will lead to the selection of data that will be most informative to the current models in training [20].

The idea of maximizing uncertainty relies on our intuition that parallel runs of CGPs can increase the diversity of the elite programs that can be exploited to guide the collection of informative data. In addition, by having parallel runs of CGP we can increase program diversity on the aggregated population level not present in the $(1+\lambda)$ EA that is most frequently used in CGP. In this study, the parallel runs are employed to estimate uncertainties only. Thus, for generating segmentation masks and all metrics related to the performance of ABS-CGP we choose from the parallel runs one program, the one with the highest fitness value on the training data.

3.1 Sampling Mechanisms

Uncertainty counts the pairwise differences between the pixels of the predicted masks for each program on an image. If two pixels have a different label between two masks, m_1 and m_2, a value of 1 is assigned to that pixel; otherwise, a value of 0 is applied. For an image of pixel dimensionality (i, j),

$$\text{uncertainty} = \sum_{i,j}^{\text{Pixels}} k = \begin{cases} 1, & \text{if } \mathrm{m}_{1(i,j)} \neq \mathrm{m}_{2(i,j)} \\ 0, & \text{otherwise.} \end{cases}$$

The Weighted Uncertainty doubles the uncertainty measurement if one pixel is labeled as 0 and the other pixel is labeled with a non-zero value. This assumes higher uncertainty should be assigned if models disagree on whether a pixel is foreground (non-zero) or background (0).

$$\text{Weighted} = \sum_{i,j}^{\text{Pixels}} k = \begin{cases} 2, & \text{if } \mathrm{m}_{1(i,j)} \neq \mathrm{m}_{2(i,j)} \text{ and } \mathrm{m}_{1(i,j)} = 0 \text{ or } \mathrm{m}_{2(i,j)} = 0, \\ 1, & \text{if } \mathrm{m}_{1(i,j)} \neq \mathrm{m}_{2(i,j)} \text{ and } \mathrm{m}_{1(i,j)} \neq 0 \text{ and } \mathrm{m}_{2(i,j)} \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Random Uniform sampling serves as a baseline. Sampling is done using uniformly distributed values to select one image at each step. There is not any information gathered from the parallel runs of CGP.

[2] (1+λ) EAs are local search algorithms and do not benefit from large populations.

3.2 Evaluation Metric and Termination Criterion

Average Precision (AP) We follow the work in [35] that defines average precision $AP = TP/(TP + FP + FN)$, where TP means true positives, FP means false positives and FN means false negatives. We use AP as our fitness function with a threshold of 0.5 to determine the true positives of the predicted mask.

Our goal is to study the effect of sampling of images from the dataset during evolution. Thus, we have a different number of images processed by ABS-CGP and standard CGP, at each generation. Given the different sizes of the datasets used during evolution, we use the number of images processed during evolution as the termination criterion. This criterion does not discriminate if there is a repetition of data points in the training set.

4 Experimental Setup

To verify if dynamically sampling the training dataset is an effective approach for CGP, we compare standard CGP with a variation of CGP that samples data during evolution, named ABS-CGP, with different sampling mechanisms. ABS-CGP builds the dataset for training using the dynamic data sampling methods (Sect. 3) and standard CGP uses all 89 images available in the training set. We run all CGP variants with the parameters shown in Table 2, the same as in [12]. ABS-CGP adds new data to the dataset for training every 100 generations, but more work is needed if we want to tune the parameters for the CGP variants, specially on the number of generations. In preliminary experiments, we found out that using 1,000,000 images processed as the budget for all CGP variants is enough for them to converge. For statistical purposes, 30 independent runs are done for each variant. We test three configurations on the number of parallel runs. For standard CGP and ABS-CGP with uniform metric, the number of parallel run is: 1, 5, 10 and 15. For the uncertainty-based metrics, the number of parallel runs is: 2, 5, 10 and 15.

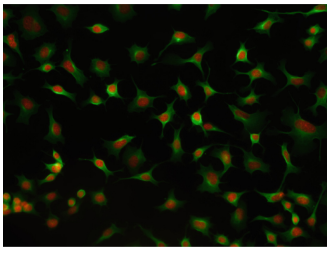


Fig. 2. A target image from the CELLPOSE dataset.

Table 2. Parameters used.

Parameter	Value
Number of nodes	30 nodes
Offspring size, λ	5
Inputs	2, α-tubulin and DAPI channels
Outputs	2, mask and markers
Mutation of function nodes	0.15
Mutation of outputs nodes	0.20
Images processed	1,000,000
Training generations	100
independent executions	30

Table 3. Mean AP performance of parallel runs of ABS-CGP with different sampling mechanisms given 30 independent executions for each configuration.

Sampling mechanism	Parallel runs	Mean (standard deviation)
Uncertainty	2	0.836 (0.018)
Uncertainty	5	0.839 (0.019)
Uncertainty	**10**	**0.840 (0.018)**
Uncertainty	15	0.837 (0.016)
Uniform	1	0.839 (0.025)
Uniform	5	0.845 (0.016)
Uniform	10	0.845 (0.017)
Uniform	**15**	**0.848 (0.014)**
Weighted uncertainty	2	0.840 (0.028)
Weighted uncertainty	5	0.840 (0.019)
Weighted uncertainty	10	0.839 (0.007)
Weighted uncertainty	**15**	**0.840 (0.015)**

4.1 Performance Comparison

For prototyping, we use the CELLPOSE dataset [35] which consists of 100 images of fluorescent-labeled protein of cultured neuroblastoma cells with phalloidin FITC and DAPI nuclear stain. For a fair comparison, we follow the work in [35], where the data is split into 89 images for training and 11 for testing. Figure 2 shows a target image from this dataset used for testing. The state-of-the-art performance in this dataset of 0.93 by DL and 0.89 with CGP [12].

4.2 Reproducibility

For reproducibility purposes, relevant data and code are available at: https://zenodo.org/records/10992287. We run the experiments on HPC resources on the OLYMPE supercomputer, a SEQUANA (ATOS-BULL) computing cluster with a power of 1.365 Pflop/s Peak, equipped with Intel Skylake 6140 processors.

5 Experimental Comparison

We systematically examine and interpret the outcomes of image segmentation into cells versus background obtained through experimentation based on overall performance and convergence speed, images sampled and number of nodes used over evolution. We highlight that all metrics and measures are calculated based on the highest-performing program of the parallel runs of the CGPs variants.

Table 3 shows the impact on performance on the test data of using parallel runs in ABS-CGP. The main result from our experiments is that data sampling

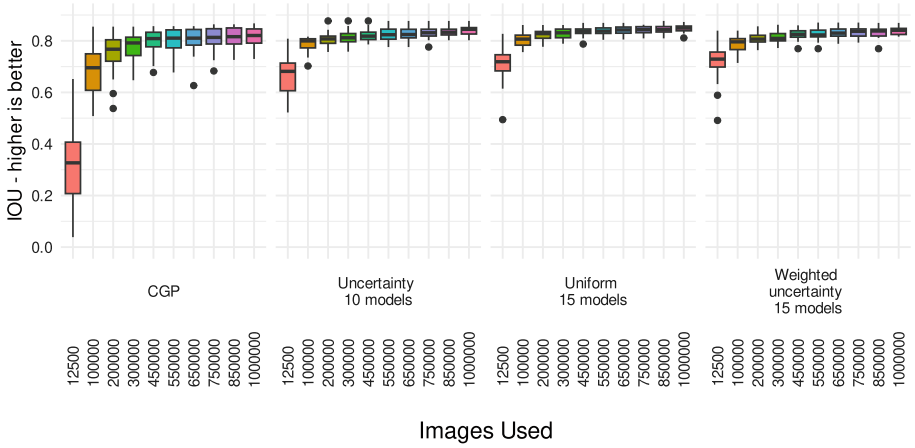


Fig. 3. Dynamically sampling images for training helps CGP to achieve high performance faster. We show the best performing configuration of CGP and ABS-CGP with: uncertainty, uniform and weighted uncertainty.

positively impacts the performance of CGP, since all methods found better performance than standard CGP that uses all data available for training, under the current experimental settings and budget. That is not a surprise, since sampling data during evolution allows the programs to explore more carefully individual features of images, without overfitting. Also, combining dynamic data sampling with parallel ABS-CGP runs increases the performance of CGP.

5.1 CGP Vs ABS-CGP

On our previous work [30], we found that using parallel runs of CGP without AL reduces the performance of CGP under the same budget, thus we compare the highest-performing ABS-CGP given each metric directly against a simple run of CGP. The highest performing ABS-CGP configurations are highlighted in **bold** in Table 3. We can verify the convergence speed of such algorithms in the sequential boxplots shown in Fig. 3. The most striking observation is that ABS-CGP clearly converges faster than traditional CGP, while all ABS-CGP showed a relatively high performance from the very beginning. To verify if there is statistically significant difference between ABS-CGP and CGP, we conducted the Mann-Withney-Wilcoxon test[3]. We highlight that we are only comparing the performance of ABS-CGP with a metric at a time against CGP, as we are not interested in performance differences between ABS-CGP variants. All ABS-CGP methods show statistical difference to CGP, with p-value < 0.05.

Figure 4 shows the number of nodes of CGP and the highest performing ABS-CGP for each sampling method. All CGP variants work with small programs

[3] Shapiro-Wilk's method showed a p-value < 0.05 implying that the distribution of the data is significantly different from normal distribution.

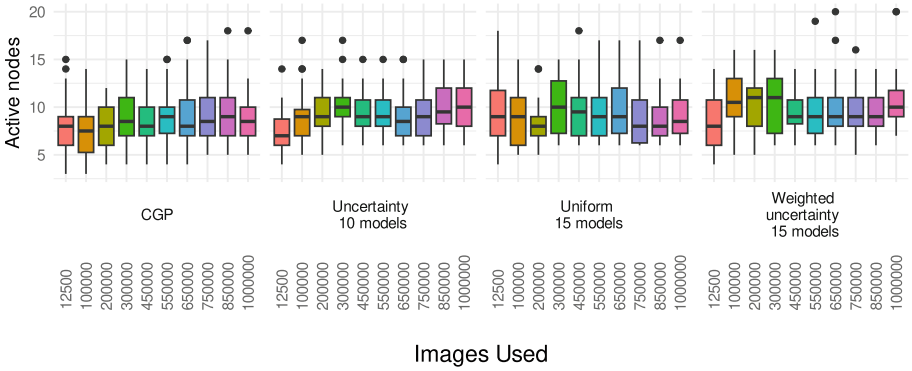


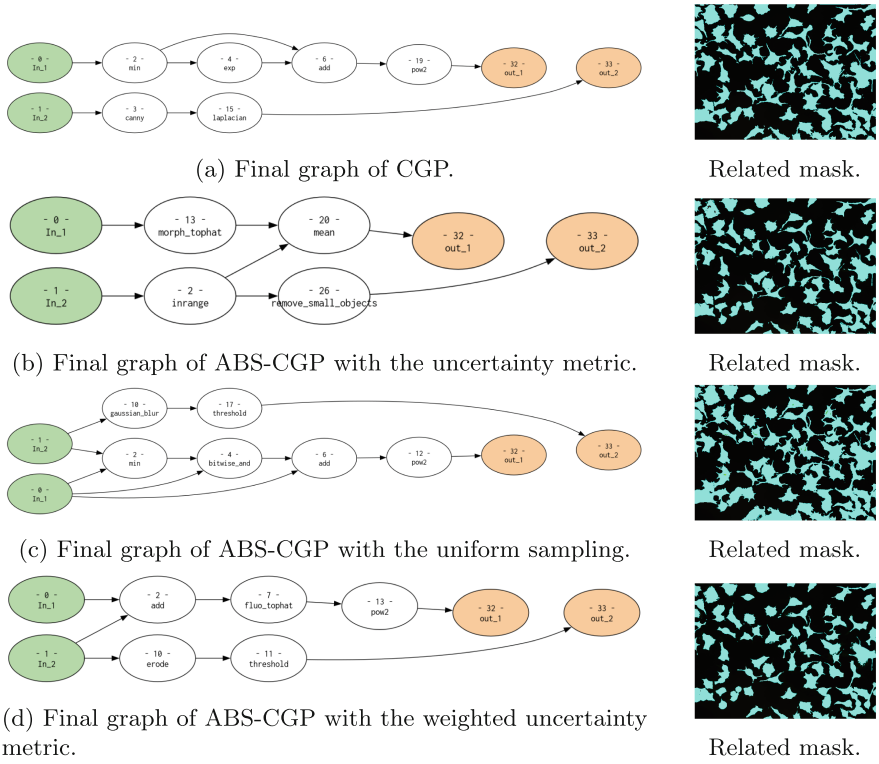
Fig. 4. Size of the programs generated given the number of active nodes.

during the whole evolution process. The size of the programs evolved by ABS-CGP varies given the sampling method, although all are working with programs slightly bigger than CGP. Figure 5 shows the final models generated by AB-CGP and the related outcome segmentation obtained using those models. The segmentation models produced by ABS-CGP show different levels of complexity and interpretability. These Figures give strong support that the models generated are overall simple and likely very interpretable.

5.2 Frequency of Images Used

The most surprising result found in this study concerns the frequency of images used. Figure 6 shows the sampling frequency of each image available to be used in the training dataset. These results are based on the aggregate outputs of ABS-CGP with uncertainty-based metrics over 60 independent executions (30 runs for each metric of these metrics, the results of traditional CGP and ABS-CGP with Uniform sampling are not shown since they do not sample images), since these metrics are the ones that capture information about images. Clearly, some images are selected more often than others.

Figure 7 show the most frequently selected images: 0, 20 and 21; and, for comparison, on and the least frequently selected ones: 78, 45 and 63. The images on the top of this Figure shows that the size, shape and brightness of the cells vary considerably, possibly indicating different details of the cells present in these images. This variation in cells indicates more complex data, which is a reasonable explanation for why these images were frequently sampled. Our analysis goes in agreement with observations from the biological field, where images with more cells are harder to analyze because the density of the cell often leads to cell membranes getting in contact, and possibly covering between cells (cells on top of each other). Additionally, images with high cell densities have a higher likelihood of showing rare events, which can make analysis more difficult. One of such events are mitosis, where a cell produces two identical nuclei in preparation for cell division, increasing cell density.



(a) Final graph of CGP. Related mask.

(b) Final graph of ABS-CGP with the uncertainty metric. Related mask.

(c) Final graph of ABS-CGP with the uniform sampling. Related mask.

(d) Final graph of ABS-CGP with the weighted uncertainty metric. Related mask.

Fig. 5. Example of interpretable graphs by ABS-CGP used to create the segmentation mask for the image in Fig. 2.

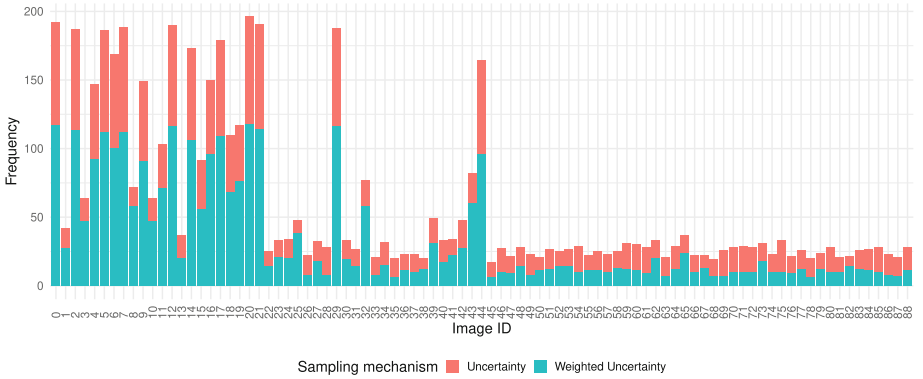
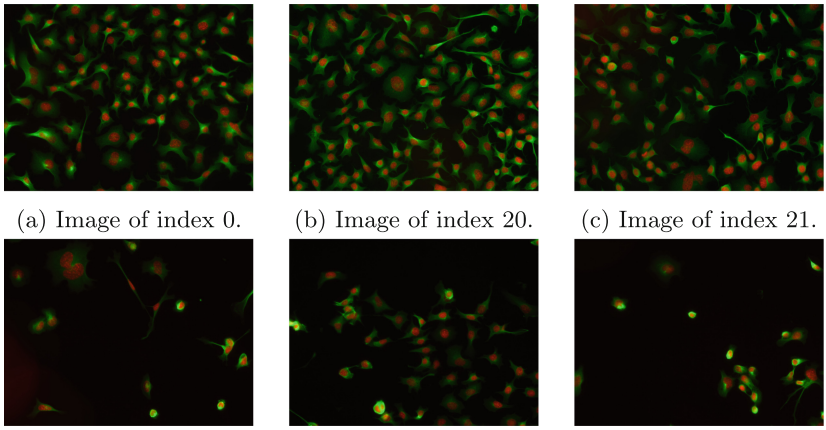


Fig. 6. Images sampled by all ABS-CGP parallel runs with both uncertainties sampling. Some images are more sampled. This suggests that there are features present in such images that lead to more uncertainty among the programs.



(a) Image of index 0. (b) Image of index 20. (c) Image of index 21.

(d) Image of index 78. (e) Image of index 45. (f) Image of index 63.

Fig. 7. Frequently sampled images (top) and least frequently sampled images (bottom). The top images lead to more uncertainty for ABS-CGP.

Looking at the bottom images of Fig. 7, however, it is possible to come up with a different scenario. The most clear distinction is that the number of cells is drastically smaller in this case, which, by itself, leads to a decrease in uncertainty values of the programs within the parallel ABS-CGP runs. This is because our metrics indirectly take into consideration the number of cells present in an image thus limiting the total uncertainty estimation of the program of the parallel ABS-CGP runs. Overall, we see several differences between the two groups of images, revealing possible insights into the characteristics of this dataset.

When we look at the scatter-plot on the image sampled frequency versus number of cells in images, Fig. 8, we see that more images are selected as the number of cells increase. Some of the most frequently selected images 0, 20 and 21 have 175, 175 and 174 cells, respectively. On the other hand, some of the least frequently selected images 78, 45 and 63 have 20, 52 and 21, respectively, a much lower number of cells. That supports our argument that the number of cells in an image leads to more uncertainty in the programs in the parallel ABS-CGP runs, however, it is clear that this alone is not the only cause of uncertainty.

5.3 CGP and Frequently Sampled Images

Given the fast convergence of the solutions shown by ABS-CGP and the preference for sampling some specific images, we test if using information about the frequency of sampled images could benefit the performance of CGP. We compare: (1) ABS-CGP with the uniform metric and 15 models, (2) the 10 most frequently sampled images ABS-CGP selected before training based on the results of the previous Sect. 5.2, (3) the 10 least frequently sampled images and (4) CGP that uses 10 images randomly selected before training.

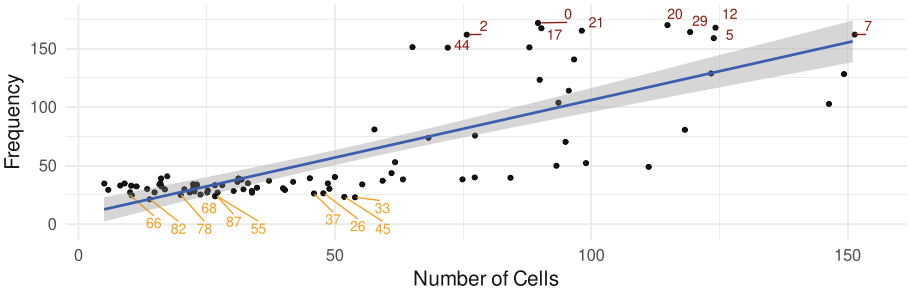


Fig. 8. Image sampled frequency versus number of cells per image. The regression line in black shows some correlation between the variables. Numbers in dark red show the index of the ten most frequently sampled images and the numbers in orange show the index of the ten least frequently sampled images. (Color figure online)

Figure 9 shows the median AP convergence results. The performance of ABS-CGP (mean: 0.85 and standard deviation: 0.01) shows that using dynamic sampling leads to high-performing models that can generalize well to the test data (as shown by the low standard deviation), suggesting we need to find a fine balance in the choice of representative images to be used in training. Interestingly, having a dataset composed of 10 images randomly selected can achieve high final results (mean: 0.83 and standard deviation: 0.03), however, with a higher standard deviation, meaning randomly selecting images before the search progress starts is sensitive to the data sampling. The CGP variants that use the most and least frequently sampled images are the worse-performing algorithms that converge early during the search process, before half of the search, showing that they are not able to generalize well to the test data. These findings exhibit the significance of dynamic sampling and Active Learning concepts in optimizing performance for this particular problem domain.

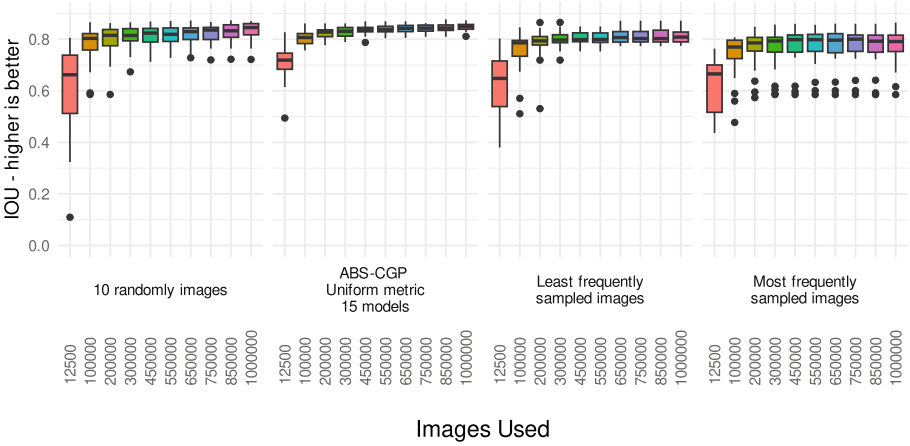


Fig. 9. Median convergence performance. ABS-CGP converges fast, to high values with lower standard deviation.

6 Discussion

We have studied how to effectively evolve programs for biomedical image segmentation using Cartesian GP (CGP). We utilized ideas from the Active Learning domain to create a training dataset that grows in size during evolution, since in we aim to sample the most informative biomedical images for model training. We found that there are two main benefits of using growing training datasets: (i) It improves the performance of the algorithms in terms of AP, and (ii) it increases the convergence speed of CGP. Additionally, we showed that having CGP trained with the 10 most frequently sampled images accelerated the convergence speed, but training CGP with a dataset composed of 10 images randomly selected also increases the convergence speed and leads better end results, indicating a subtle balance in the choice of images for training. Later, to provide a more comprehensive analysis of ABS-CGP we will apply this method to different datasets and we compare it against Lexicase selection [21,27].

Our results also show that the data sampling in CGP provides valuable insights into the features that induce uncertainty in the programs. For example, we found a group of images that are frequently sampled, and that these images contain many cells that vary in size, shape and color. Through systematic observation and further analysis of the specific features of these images, we might be able to reduce undesired variability in program output. This analysis might help increase the interpretability of programs and also lead to additional improvements in performance. The ideas explored in this contribution can serve as a basis for future research on the co-evolution of datasets. By exploring the evolution of datasets in connection with the programs generated from them, CGP might find more general programs that can deal efficiently with more diverse data sets that better capture the complexities of real-world scenarios.

Finally, this work represents a new stride towards achieving a broader objective in the construction of an interactive learning system within the domain of biomedical image analysis. We expect that the combination of CGP with Active Learning has the potential to reinforce the development of applications where a human expert annotates specific images or areas of large images upon request. The images or areas for annotation would be suggested by the learning algorithm based on its current requirement for annotated data, such as specific cell types, color diversity, shapes, or others. The ultimate aim is to foster a collaborative effort between our CGP learner and a human expert, guiding the learning procedure through the complexities of the images to be analyzed.

Acknowledgments. This project used compute resources from the CALMIP project P21049. This work is funded by the Laboratoire d'Excellence Toulouse Cancer TOUCAN, contract ANR11-LABX. This work is supported by the AI Interdisciplinary Institute ANITI, funded by the French program "Investing for the Future - PIA3" under Grant agreement no. ANR-19-PI3A-0004.

References

1. Ahmad, A.M., Khan, G.M., Mahmud, S.A., Miller, J.F.: Breast cancer detection using Cartesian genetic programming evolved artificial neural networks. In: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation (GECCO 2012), pp. 1031–1038. Association for Computing Machinery, New York (2012). https://doi.org/10.1145/2330163.2330307
2. Ain, Q.U., Al-Sahaf, H., Xue, B., Zhang, M.: Genetic programming for automatic skin cancer image classification. Expert Syst. Appl. **197**, 116680 (2022). https://doi.org/10.1016/j.eswa.2022.116680
3. Ain, Q.U., Al-Sahaf, H., Xue, B., Zhang, M.: Automatically diagnosing skin cancers from multimodality images using two-stage genetic programming. IEEE Trans. Cybernet. **53**(5), 2727–2740 (2023). https://doi.org/10.1109/TCYB.2022.3182474
4. Ain, Q.U., Xue, B., Al-Sahaf, H., Zhang, M.: Genetic programming for multiple feature construction in skin cancer image classification. In: 2019 International Conference on Image and Vision Computing New Zealand (IVCNZ), pp. 1–6. Institute of Electrical and Electronics Engineers (IEEE), Piscataway (2019). https://doi.org/10.1109/IVCNZ48456.2019.8961001
5. Alkhaldi, E., Salari, E.: Ensemble optimization for invasive ductal carcinoma (IDC) classification using differential Cartesian genetic programming. IEEE Access **10**, 128790–128799 (2022). https://doi.org/10.1109/ACCESS.2022.3228176
6. Ben Hamida, S., Hmida, H., Borgi, A., Rukoz, M.: Adaptive sampling for active learning with genetic programming. Cognit. Syst. Res. **65**, 23–39 (2021).https://doi.org/10.1016/j.cogsys.2020.08.008
7. Beucher, S., Meyer, F.: The morphological approach to segmentation: the watershed transformation. In: Mathematical Morphology in Image Processing, pp. 433–481. CRC Press (2018)
8. Bi, Y., Xue, B., Zhang, M.: Genetic programming-based discriminative feature learning for low-quality image classification. IEEE Trans. Cybernet. **52**(8), 8272–8285 (2022). https://doi.org/10.1109/TCYB.2021.3049778
9. Biau, J., Wilson, D., Cussat-Blanc, S., Luga, H.: Improving image filters with Cartesian genetic programming. In: IJCCI, pp. 17–27 (2021)
10. Casanova, A., Pinheiro, P.O., Rostamzadeh, N., Pal, C.J.: Reinforced active learning for image segmentation. In: International Conference on Learning Representations (2020). https://openreview.net/forum?id=SkgC6TNFvr
11. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. J. Artif. Intell. Res. **4**, 129–145 (1996). https://doi.org/10.1613/jair.295
12. Cortacero, K., et al.: Evolutionary design of explainable algorithms for biomedical image segmentation. Nat. Commun. **14**(1), 7112 (2023)
13. Csiba, D., Richtárik, P.: Importance sampling for minibatches. arXiv preprint arXiv:1602.02283 (2016). https://api.semanticscholar.org/CorpusID:15471954
14. Deng, S., et al.: Deep learning in digital pathology image analysis: a survey. Front. Med. **14**, 470–487 (2020)
15. Esteva, A., et al.: Deep learning-enabled medical computer vision. NPJ Digit. Med. **4**(1), 5 (2021)
16. Gaillochet, M., Desrosiers, C., Lombaert, H.: Active learning for medical image segmentation with stochastic batches. Med. Image Anal. **90**,102958 (2023). https://doi.org/10.1016/j.media.2023.102958
17. Gathercole, C., Ross, P.: Dynamic training subset selection for supervised learning in Genetic Programming. In: Davidor, Y., Schwefel, H.-P., Männer, R. (eds.) PPSN

1994. LNCS, vol. 866, pp. 312–321. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-58484-6_275
18. Harding, S., Leitner, J., Schmidhuber, J.: Cartesian Genetic Programming for Image Processing, pp. 31–44. Springer, New York (2013). https://doi.org/10.1007/978-1-4614-6846-2_3
19. Haut, N., Banzhaf, W., Punch, B.: Active learning improves performance on symbolic regression tasks in StackGP. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO 2022), pp. 550–553. Association for Computing Machinery, New York (2022). https://doi.org/10.1145/3520304.3528941
20. Haut, N., Punch, B., Banzhaf, W.: Active learning informs symbolic regression model development in genetic programming. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation (GECCO 2023), pp. 587–590. Companion, Association for Computing Machinery, New York (2023). https://doi.org/10.1145/3583133.3590577
21. Helmuth, T., Spector, L., Matheson, J.: Solving uncompromising problems with lexicase selection. IEEE Trans. Evol. Comput. **19**(5), 630–643 (2015). https://doi.org/10.1109/TEVC.2014.2362729
22. Hmida, H., Hamida, S.B., Borgi, A., Rukoz, M.: Sampling methods in genetic programming learners from large datasets: a comparative study. In: Angelov, P., Manolopoulos, Y., Iliadis, L., Roy, A., Vellasco, M. (eds.) INNS 2016. AISC, vol. 529, pp. 50–60. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-47898-2_6
23. Hüllermeier, E.: How to measure uncertainty in uncertainty sampling for active learning. Mach. Learn. **111**, 89–122 (2021). https://api.semanticscholar.org/CorpusID:221726816
24. Huml, M., Silye, R., Zauner, G., Hutterer, S., Schilcher, K., et al.: Brain tumor classification using AFM in combination with data mining techniques. BioMed Res. Int. **2013**, 1–11 (2013)
25. Khan, A., Qureshi, A.S., Wahab, N., Hussain, M., Hamza, M.Y.: A recent survey on the applications of genetic programming in image processing. Comput. Intell. **37**(4), 1745–1778 (2021).https://doi.org/10.1111/coin.12459
26. Kirsch, A., van Amersfoort, J., Gal, Y.: BatchBALD: efficient and diverse batch acquisition for deep Bayesian active learning. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox,E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019). https://proceedings.neurips.cc/paper_files/paper/2019/file/95323660ed2124450caaac2c46b5ed90-Paper.pdf
27. La Cava, W., Spector, L., Danai, K.: Epsilon-lexicase selection for regression. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016 (GECCO 2016), pp. 741–748. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2908812.2908898
28. Lavinas, Y., Cortacero, K., Cussat-Blanc, S.: Evolving graphs with Cartesian genetic programming with lexicase selection. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation (GECCO 2023), pp. 1920–1924. Association for Computing Machinery, New York (2023). https://doi.org/10.1145/3583133.3596402
29. Lavinas, Y., Haut, N., Punch, W., Banzhaf, W., Cussat-Blanc, S.: Data sampling via active learning in Cartesian genetic programming for biomedical data. In: 2024 IEEE Congress on Evolutionary Computation (CEC). p. To Appear (2024)

30. Lavinas, Y., Haut, N., Punch, W., Banzhaf, W., Cussat-Blanc, S.: Dynamically sampling biomedical images for genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO 2024). p. To Appear. Association for Computing Machinery, New York (2024). https://doi.org/10.1145/3638530.3654202
31. Miller, J.F.: An empirical study of the efficiency of learning Boolean functions using a Cartesian genetic programming approach. In: Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation (GECCO 1999), vol. 2, pp. 1135–1142. Morgan Kaufmann Publishers Inc., San Francisco (1999)
32. Nath, V., Yang, D., Landman, B.A., Xu, D., Roth, H.R.: Diminishing uncertainty within the training pool: active learning for medical image segmentation. IEEE Trans. Med. Imaging **40**(10), 2534–2547 (2021). https://doi.org/10.1109/TMI.2020.3048055
33. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)
34. Settles, B.: Active learning literature survey. In: Computer Sciences Technical Report 1648. University of Wisconsin, Madison (2009)
35. Stringer, C., Wang, T., Michaelos, M., Pachitariu, M.: Cellpose: a generalist algorithm for cellular segmentation. Nat. Methods **18**(1), 100–106 (2021)
36. Suganuma, M., Shirakawa, S., Nagao, T.: Designing convolutional neural network architectures using Cartesian genetic programming, pp. 185–208. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-3685-4_7
37. Tokdar, S.T., Kass, R.E.: Importance sampling: a review. Wiley Interdiscip. Rev.: Comput. Statist. **2** (2010). https://api.semanticscholar.org/CorpusID:14552197
38. Vanneschi, L., Farinaccio, A., Giacobini, M., Mauri, G., Antoniotti, M., Provero, P.: Identification of individualized feature combinations for survival prediction in breast cancer: a comparison of machine learning techniques. In: Pizzuti, C., Ritchie, M.D., Giacobini, M. (eds.) EvoBIO 2010. LNCS, vol. 6023, pp. 110–121. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12211-8_10
39. Vladislavleva, E., Smits, G., den Hertog, D.: On the importance of data balancing for symbolic regression. IEEE Trans. Evol. Comput. **14**(2), 252–277 (2010). https://doi.org/10.1109/TEVC.2009.2029697
40. Wilson, D.G., Cussat-Blanc, S., Luga, H., Miller, J.F.: Evolving simple programs for playing Atari games. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2018), pp. 229–236. Association for Computing Machinery, New York (2018). https://doi.org/10.1145/3205455.3205578
41. Xia, T., Cosgrove, J., Alty, J., Jamieson, S., Smith, S.: Application of classification for figure copying test in Parkinson's disease diagnosis by using Cartesian genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO 2019), pp. 1855–1863. Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3319619.3326822